## IN THE CLAIMS

12. (Amended) A method for operating a pipelined microprocessor, said method comprising:

detecting in said pipelined microprocessor an instruction that loads data from a first memory location that was previously stored to without computing an external memory address of said first memory location.

13. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 12, said method further comprising:

detecting in said pipelined microprocessor an instruction that stores data into a second memory location that was previously read from without computing an external memory address of said second memory location.

14. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 12, said method further comprising:

detecting in said pipelined microprocessor instructions that load data from identical memory locations that were previously stored to without computing external memory addresses of said identical memory locations.

et 01

15. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 13, said method further comprising:

detecting in said pipelined microprocessor instructions that store data into identical memory locations that were previously read from without computing external memory addresses of said identical memory locations.

16. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 14, said method further comprising:

examining in said pipelined microprocessor symbolic structure of said instructions that load data from identical memory locations that were previously stored to; and

detecting said instructions that load data from identical memory locations by examining said symbolic structure.

17. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 15, said method further comprising:

examining in said pipelined microprocessor symbolic structure of said instructions that store data into identical memory locations that were previously read from; and

detecting said instructions that store data into identical memory locations by examining said symbolic structure.

C401

18. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 16, said method further comprising:

detecting in an instruction decode stage of said pipelined microprocessor said instructions that load data from identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

sending a bypass signal from a bypass element to an instruction execution stage of said pipelined microprocessor wherein said bypass signal indicates that said instructions refer to an identical memory location.

19. (Amended) A method for operating a pipelined microprocessor as claimed in Claim 17, said method further comprising:

detecting in an instruction decode stage of said pipelined microprocessor said instructions that store data into identical memory locations by identifying an identical offset address value from an identical base address value in a register within said pipelined microprocessor; and

sending a bypass signal from a bypass element to an instruction execution stage of said pipelined microprocessor wherein said bypass signal indicates that said instructions refer to an identical memory location.

CTD1

20. (Amended) A method for operating a pipelined microprocessor, said method comprising:

detecting a first instruction that stores data to a first memory location, said first instruction comprising syntax for computing an effective address for said first memory location;

detecting a second instruction that loads data from a second memory location, said second instruction comprising syntax for computing an effective address for said second memory location;

determining said syntax for said first instruction and said syntax for said second instruction;

using said syntax for said first instruction and said syntax for said second instruction to determine a relationship between said first memory location and said second memory location, without computing said effective address for said first memory location and without computing said effective address for said second memory location; and

using said relationship to determine whether to perform one of said first instruction and said second instruction.